

REMARKS

Introduction

Claims 1-15 were pending. Claim 1-3, and 14-15 have been amended hereby. Claims 1, 14, and 15 are independent. Claim 11 has been cancelled. Entry of this Amendment, and reconsideration of the above-identified application in view of the following remarks, is respectfully requested.

Rejections under 35 U.S.C. § 112

Claims 1-15 stand rejected under 35 U.S.C. 112, second paragraph because the Examiner asserts that it is not clear whether the “operation” and the “nested operation” of claims 1, 14, and 15 are the same or distinct operations. Claims 1, 14, and 15 have been amended to recite “a first operation” and “a nested operation within said first operation.” Thus, it can be seen that the two operations are distinct but related. Withdrawal of the 35 U.S.C. 112 rejection to Claims 1-15 is respectfully requested.

Rejections under 35 U.S.C. § 103(a)

Claims 1-15 stand rejected under 35 U.S.C. 103(a) as unpatentable over U.S. Patent Application Publication No. 20030140332 (Norton) in view of U.S. Patent Application Publication No. 20040073870 (Fuh) and further in view of U. S. Patent No. 5,261,095 (Crawford).

Amended claim 1 of the present application recites, *inter alia*, a method for use in a distributed processing system to specify an application service comprising: defining a schema written in XML comprising a first operation having a plurality of arguments, the schema having

a nested operation within said first operation; validating the first operation's signature using said schema; executing said first operation on a first processor in said distributed processing system **in real time**; sending said nested operation to a second processor in the distributed processing system; and executing said nested operation on said second processor in said distributed processing system in real time, wherein said schema renders unaltered underlying function calls which define said first operation and said nested operation. Amended claim 15 is similar scope to amended claim 1, except for the additional limitation that the first operation is executed at a first service application on a first processor, and the nested operation is executed at a second service application on said second processor, with the first service becoming a client application for the second service application. Amended claim 15 does not contain the wherein clause pertaining to the schema leaving the underlying function calls unaltered.

Thus, by amended claim 1 and 15, a method is provided to specify new functionality that can relate to distributed data processing systems. The present invention defines a schema which encodes an existing program that resides on one computer, which is composed of operation or function or method calls, which has, as described in the specification of the present application as filed at page 2, lines. 3-5, "a signature, i.e., a number or a number and data type of various parameters that must be passed to the function, and a return type (*i.e.*, the meaning that is typically employed when dealing with traditional programming languages, such as C, C++ or Java)". The parameters may themselves be function calls or operations, which create a nested function. Several of these function calls are grouped together into a program. The function call containing the nested function call or calls is executed on one processor **in real time** and the nested function calls are executed on a second processor in the distributed processing system, i.e., the operations of the function calls can be distributed among several processors to be executed simultaneously or in series **in real time**. The processors can be co-

resident on the same computer or distributed within a network such as the Internet. On each processor, the function signature must be checked for validity using the schema, i.e., whether the signature matches what it determines from its stored function of the same name is a valid signature. Support for execution **in real time** can be found at least at paragraph [0016] of U.S. Patent Application Publication No. US 2004/0230587 A1 (the present application as published).

Norton describes a system for generating a specialized software development tool, and uses a definition file that defines actions to be performed by the tool, a schema that defines inputs for the tool, and a resource file, that includes information required by the tool at runtime, to generate the software development tool. The tool is used to automatically generate a program. (See Norton at abstract and FIG. 1.). The system of Norton is a program for generating a program. At page 2, paragraph [0021], Norton states that “the present invention may be implemented on virtually any type of computer,” but shows this computer in FIG. 1 as “a typical computer (10) includes a processor (12).” This implies that the program of Norton is not executed on multiple processors in a distributed processing system. Furthermore, at paragraphs [0044] and [0045], Norton describes the software development tool 34 as being run in a stand alone runtime environment or an integrated runtime environment, which implies a single processor executing the program **which is compiled off line and then executed at run time**, not executing said operation **on a first processor** in said distributed processing system on the fly **in real time**; and executing said nested operation **on a second processor** in said distributed processing system on the fly **in real time** as recited by amended claims 1 and 15. Accordingly, Norton fails to teach or suggest each and every limitation of claims 1 and 15 as amended.

Thus, Norton does not describe a program being executed on multiple processors, nor does Norton describe executing said operation **on a first processor** in said distributed

processing system in **real time**; and executing said nested operation **on a second processor in real time** in said distributed processing system as recited by amended claim 1.

Fuh fails to correct the deficiencies of Norton. Fuh describes a system that includes an XML schema compiler front end, an XML schema compiler back end, and an XML document runtime validation engine, where the runtime validation engine includes an XML schema validation parser. An annotated automaton encoding (AAE) for an XML schema definition after compilation is loaded into the XML schema validation parser; and the XML document is validated against the XML schema definition by the XML schema validation parser utilizing the annotated automaton encoding. (See Fuh at abstract and FIG. 1.) Fuh describes that each of different XML schema definitions are compiled once into AAE (Annotated Automaton Encoding) format. The AAE parser can then be used on any AAE format definition set.

Nowhere in Fuh is it mentioned as to where the system is executed: either on one processor or more, stand alone or distributed environment. The XML documents are first compiled off line and then put into a run time validation parser. There is no mention of an operation and a nested operation being executed on **different processors in real time**. Accordingly, Fuh fails to teach or suggest each and every limitation of amended claims 1 and 15, as amended, namely: executing said operation **on a first processor in real time** in said distributed processing system; and executing said nested operation **on a second processor in real time** as recited by amended claims 1 and 15.

Crawford et al. fails to correct the deficiencies of Norton and Fuh. Crawford discloses a method of partitioning a computer software program, so that a main program is executable by a first processor, and at least one designated subprogram may be executed by a second processor. The user selects a subprogram to be executed by the second processor and assigns it an identifier that is global to both processors. The original program is modified to create a main program,

which has an argument passing means that creates a software environment common to both processors, and a subprogram capable of accepting the parameters from the main program and executing its function. The subprogram's call in the main program is converted into a packet form that provides the function identification and argument passing means. By compiling the subprogram for the second processor, it becomes executable on the second processor.

Crawford et al. does disclose a means for partitioning a program so that a main program is executable on one processor and one or more function calls within the main program on one or more other processors. However, the program is partitioned and modified off-line to put the main program and function calls into a form that separate processors can execute their respective portions. These separate portions are then compiled off line, then executed in run time environments. Nowhere in Crawford et al. is it disclosed or taught that the separate main function and internal functions are executed on the fly **in real time** as recited in amended claims 1 and 15.

Furthermore, neither Norton nor Fuh nor Crawford et al., alone or in combination, teach or disclose **wherein said schema renders unaltered underlying function calls which define said first operation and said nested operation.** Fuh is agnostic about underlying function calls being unaltered by the signature verification process, or by the front and back end compilers. Norton is a software development tool code generator and thus by definition creates new function calls different from the input program. Crawford et al. specifically states that both the main function and a second function to be executed on another processor are altered before compile time by rendering the signatures global to each program using macros and #defines, so that the code that is to be executed is different from the original program. This teaches away from the present claimed invention, as shown by the XML code of the figures, which has the schema encoding the signatures of underlying function calls of the first operation and the nested

operation to XML without altering the original function calls from which they originate. This also makes it possible to execute the function calls on various processors on the fly **in real time** without extra programming steps.

Furthermore, neither Norton nor Fuh nor Crawford et al., alone or in combination, teach or disclose executing the first operation **at a first service application**, and executing said nested operation **at a second service application**, wherein the **first service application becomes a client application for the second service application** after executing the first operation and then requesting execution of the nested operation on a second processor as taught by amended independent claim 15. In the claimed invention, the operation can be executed locally on one service application, while the nested operation can be passed to a subsequent service operation, wherein the current service application becomes the client for a secondary service application. In any of Norton, Fuh, or Crawford et al., there is no concept of client and server applications. Norton and Fuh are stand-alone applications, while Crawford et al. just partitions applications as running separate programs, with no client and no server architectural elements. The client constructs a message as needed and passes the operation along to the next service processor. (see the present application as published as US 2004/0230587 A1, page 1, paragraphs [0015] and [0016]). As discussed above, there is no execution of one operation as a first service application on one processor and executing another operation on another service processor, the **first service application becoming a client application for the second service application** in either Norton, Fuh, or Crawford et al.

As such, the Applicants respectfully request that the §103(a) rejection of claims 1 and 15 and claims dependent therefrom (i.e., Claims 2-13) be withdrawn.

Furthermore, neither Norton nor Fuh nor Crawford et al., alone or in combination, teach or disclose each and every limitation of amended independent claim 14, namely “using

said schema to generate a second program having an interface with **grid management software**, the second program and the grid management software being used to render the nested operation executable on said second processor.” Support for **grid management software**, including its definition, can be found at least at paragraph [0017] of the present application as published.

Norton and Crawford et al. do describe code generation, but there is no mention of the code generate being compatible with grid management software for partitioning a program to be executed on various processors in a grid of processors. As such, the Applicants respectfully requests that claim 14 be allowed.

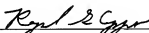
Thus, applicant submits that each of the claims of the present application are patentable over each of the references of record, either taken alone, or in any proposed hypothetical combination. Accordingly, withdrawal of the rejections to the claims is respectfully requested.

Conclusion

In view of the above remarks, reconsideration and allowance of the present application is respectfully requested. If any additional fee is deemed necessary, then the Commissioner is authorized to charge such fee to Deposit Account No. 50-1358. Applicant's undersigned patent agent may be reached by telephone at (973) 597-2500. All correspondence should continue to be directed to our address listed below.

Respectfully submitted,

Date: 7/30/07


Raymond G. Cappo
Patent Agent
Registration No. 53,836

DOCKET ADMINISTRATOR
LOWENSTEIN SANDLER PC
65 Livingston Avenue
Roseland, NJ 07068